Classifying Web Pages as Evergreen or Non - Evergreen

1. Problem Statement : Stumble Upon Evergreen Classification **,** the goal is to build a classifier to categorize a web page as *evergreen* (long time relevance) or *non-evergreen* (short time relevance)

2. Given dataset:

- train.tsv: is the training set containing 7,395 observations and 26 features and Binary evergreen labels as Evergreen (1) or Non-Evergreen (0).
- test.tsv: is the test/evaluation set containing 3,171 observations and 26 features.

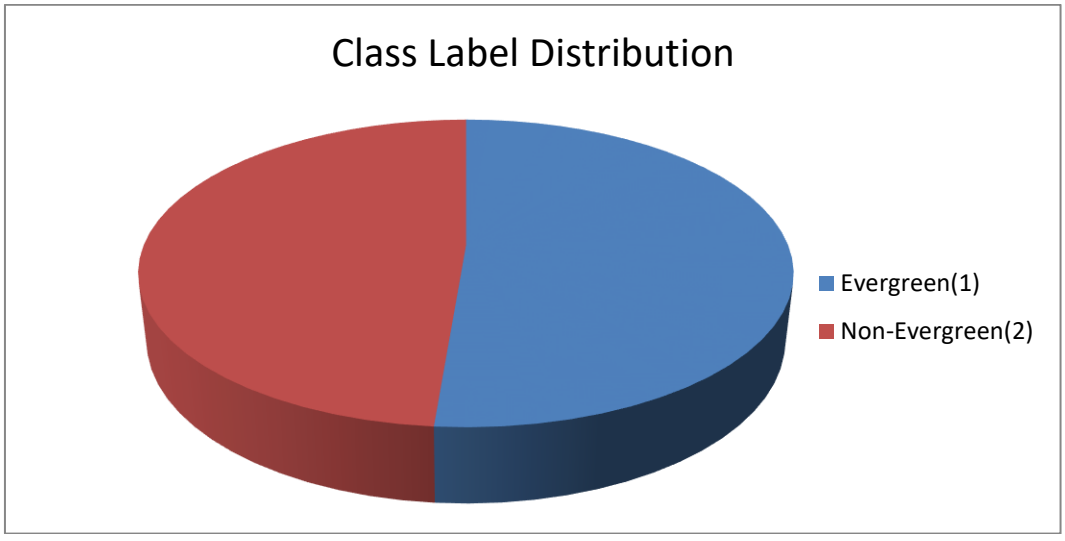
Note: some features are in text form, categorical form and numeric form.

3. Approach:

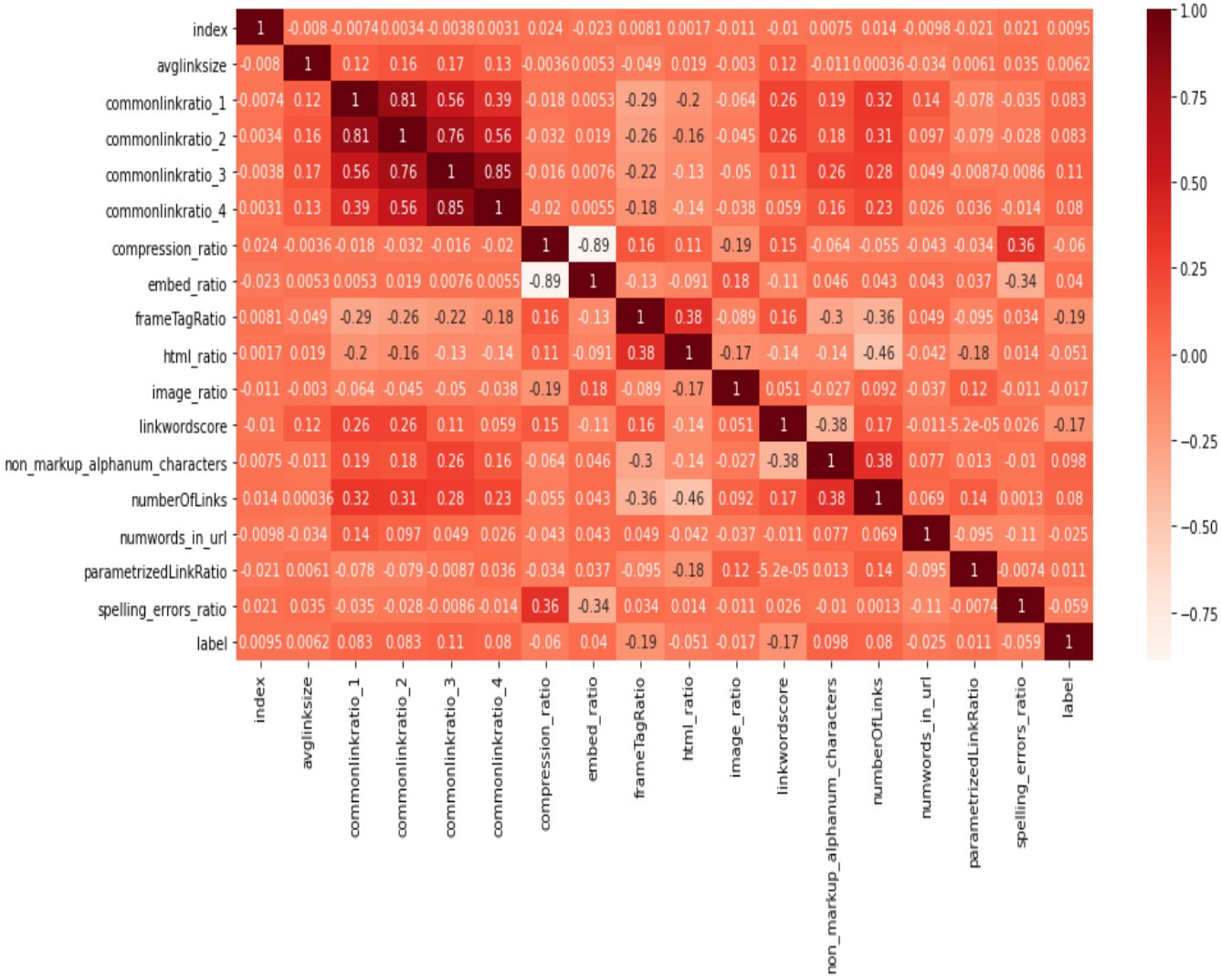
- Data preprocessing
- Model Building
- Result

a.) Data preprocessing: Cleaning and Encoding text dataset Retrieve text data from .json file i.e. title, body and url.

i. Replace all non-alphabetic and special characters to space.
ii. Lower the case
iii. **Tokenization**
iv. **Lemmatization**
v. Prepare list of lists that conation lemmatize words of sentence
vi. Prepare vocabulary
vii. **One-Hot Encoding** on words
viii. Add **Padding** (to make same length i.e. max sentence length).
ix. Use pre-trained **Word Embedding Model** for representation of word that allows words with similar meaning to have a similar representation (semantic meaning).
x. Preparing Numeric dataset: Check null values and replace null values with mean of values corresponding to class label.
xi. Check data is balanced or not:
Number of Observation in class label(1) - 3796
Number of Observation in class label(0) - 3599.



Correlation b/w Numeric Independent feature and Target Labels



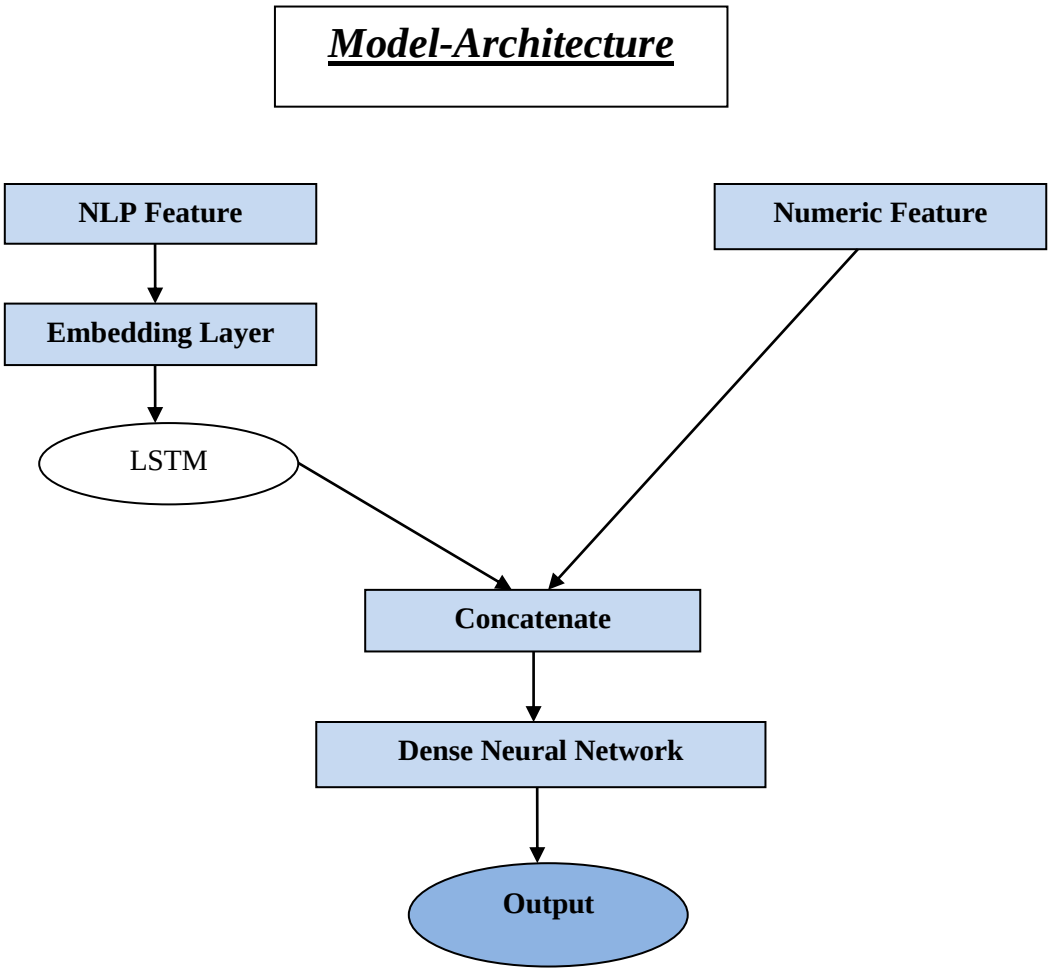
+

Note:

1. Calculate correlation between “Numerical Independent Feature” and “Target Label”.
2. Take only those features which have greater than 0.1 Correlation.
3. At Last concatenate text preprocessed data or numeric data.

b.) ***Model Building :***

Use **LSTM** to store long and short term dependency between data. Prepare Dense Neural-N/W with “**Relu”** activation function ,**Cross-entropy** loss function and compile the model using **Adam optimizer** with **learning-rate** = 0.3.



c.) ***Result :***

Neural N/W Architecture 1 : 63/70/60/40/40/40/50/20
Accuracy : 67%
Recall = 55%
Precision = 75%

Neural N/W Architecture 2 : 63/70/80/90/100/110
Accuracy : 65%
Recall = 74%
Precision = 65%

4. Conclusion :

If we see from Classification point of view then Precision and Recall both are important but when we use this Model for Recommendation purpose then Precision is given higher importance than Recall reason being that, in case of recommendation system if we do not recommend sites which are of viewer choice of interest then also it is fine because a viewer can find many other interests but if we recommend site which viewer doesn't like then this can bother him.